php

Search

urlencode

Change language: French

Edit Report a Bug

(PHP 4, PHP 5)
urlencode — Encode une chaîne en URL

Description

```
string urlencode ( string $str )
```

Cette fonction est utile lors de l'encodage d'une chaîne de caractères à utiliser dans la partie d'une URL, comme façon simple de passer des variables vers la page suivante.

Liste de paramètres

str
La chaîne de caractères à encoder.

Valeurs de retour

Retourne une chaîne dont les caractères non alphanumériques (hormis -_.) sont remplacés par des séquences commençant par un caractère pourcentage (*%*), suivi de deux chiffres hexadécimaux. Les espaces sont remplacés par des signes plus (*+*). Ce codage est celui qui est utilisé pour poster des informations dans les formulaires HTML. Le type MIME est *application/x-www-form-urlencoded*. Ce codage est différent de celui spécifié dans la » RFC 3986 (voir rawurlencode()) : pour des raisons historiques, les espaces sont remplacés par des signes plus (*+*).

Exemples

Exemple #1 Exemple avec urlencode()

```php
<?php
echo '<a href="mycgi?foo=', urlencode($userinput), '">';
?>
```

Exemple #2 Exemple avec urlencode() et htmlentities()

```php
<?php
$query_string = 'foo=' . urlencode($foo) . '&bar=' . urlencode($bar);
echo '<a href="mycgi?' . htmlentities($query_string) . '">';
?>
```

Notes

Note:
Faites bien attention aux variables qui ressemblent à des entités HTML, comme &, © et £, qui sont analysées par le client web et remplacées par leur valeur. C'est un vrai problème qui a été montré par le W3C depuis longtemps. La référence est ici : » http://www.w3.org/TR/html4/appendix/notes.html#h-B.2.2.
PHP supporte le remplacement de séparateur d'arguments par un point-virgule, comme recommandé par le W3C, grâce à la directive **arg_separator** du fichier `php.ini`. Malheureusement, la plupart des clients web n'envoient pas leurs données de formulaire avec des points-virgules. Une solution plus portable est d'utiliser *&* à la place de *&* comme séparateur. Vous n'avez alors pas à changer la directive **arg_separator** . Laissez-la à &, mais encodez vos URL en utilisant htmlentities() ou htmlspecialchars().

Voir aussi

- urldecode() - Décode une chaîne encodée URL
- htmlentities() - Convertit tous les caractères éligibles en entités HTML
- rawurlencode() - Encode une chaîne en URL, selon la RFC 3986
- rawurldecode() - Décode une chaîne URL
- » RFC 3986

User Contributed Notes 30 notes

add a note

▲ 5 ▼ edwardzyang at thewritingpot dot com 9 years ago

I was testing my input sanitation with some strange character entities. Ones like ⍰ and ⍰ were passed correctly and were in their raw form when I passed them through without any filtering.

However, some weird things happen when dealing with characters like (these are HTML entities): ‼ ▐ ┐and Θ have weird things going on.

If you try to pass one in Internet Explorer, IE will *disable* the submit button. Firefox, however, does something weirder: it will convert it to it's HTML entity. It will display properly, but only when you don't convert entities.

The point? Be careful with decorative characters.

PS: If you try copy/pasting one of these characters to a TXT file, it will translate to a ?.

▲ 6 ▼ Thomas 6 years ago

Reply to 'peter at mailinator dot com'

If you are having problems using urldecode in PHP following the escape() function in Javascript, try to do a decodeURI() before the escape(). This fixed it for me at least.

Thomas

▲ 9 ▼ ahrensberg at gmail dot com 6 years ago

Like "Benjamin dot Bruno at web dot de" earlier has writen, you can have problems with encode strings with special characters to flash. Benjamin write that:

```
<?php
  function flash_encode ($input)
```

```
  {
    return rawurlencode(utf8_encode($input));
  }
?>
```

... could do the problem. Unfortunately flash still have problems with read some quotations, but with this one:

```
<?php
  function flash_encode($string)
  {
    $string = rawurlencode(utf8_encode($string));

    $string = str_replace("%C2%96", "-", $string);
    $string = str_replace("%C2%91", "%27", $string);
    $string = str_replace("%C2%92", "%27", $string);
    $string = str_replace("%C2%82", "%27", $string);
    $string = str_replace("%C2%93", "%22", $string);
    $string = str_replace("%C2%94", "%22", $string);
    $string = str_replace("%C2%84", "%22", $string);
    $string = str_replace("%C2%8B", "%C2%AB", $string);
    $string = str_replace("%C2%9B", "%C2%BB", $string);

    return $string;
  }
?>
```

... should solve this problem.

▲ 4 ▼ omid at omidsakhi dot com 4 years ago

I needed a function in PHP to do the same job as the complete escape function in Javascript. It took me some time not to find it. But findaly I decided to write my own code. So just to save time:

```
<?php
function fullescape($in)
{
 $out = '';
 for ($i=0;$i<strlen($in);$i++)
 {
  $hex = dechex(ord($in[$i]));
  if ($hex=='')
    $out = $out.urlencode($in[$i]);
  else
    $out = $out .'%'.((strlen($hex)==1) ? ('0'.strtoupper($hex)):(strtoupper($hex)));
 }
 $out = str_replace('+','%20',$out);
 $out = str_replace('_','%5F',$out);
 $out = str_replace('.','%2E',$out);
 $out = str_replace('-','%2D',$out);
 return $out;
}
?>
```

It can be fully decoded using the unscape function in Javascript.

▲ 4 ▼ daniel+php at danielnorton dot com 4 years ago

Don't use urlencode() or urldecode() if the text includes an email address, as it destroys the "+" character, a perfectly valid email address

character.

Unless you're certain that you won't be encoding email addresses AND you need the readability provided by the non-standard "+" usage, instead always use use rawurlencode() or rawurldecode().

▲ 7 ▼ leaksin [at] gmail [dot] com 1 year ago

"cleaning the URL",Totally and in a nut shell:
1.You must use rawurlencode() for parts that come before "?"
2.Use urlencode for all GET parameters (values that come after each "=")(POST parameters are automatically encoded).
3.Use htmlspecialchars for HTML tag parameters and HTML text content

```
<?php
$url_page = 'example/page/url.php';
//page the link will request
$text = 'this is a simple string';
$id = '4334%3434';
$linktext = "<Clickit> & you will see it";
//text of the link, with HTML unfriendly characters
?>
<?php
// this gives you a clean link to use
$url = "http://localhost/";
$url .= rawurlencode($url_page);
$url .= "?text=" . urlencode($text);
$url .= "&id=" . urlencode($id);

// htmlspecialchars escapes any html that
// might do bad things to your html page
?>
<a href="<?php echo htmlspecialchars($url); ?>">
<?php echo htmlspecialchars($linktext); ?>
</a>
```

▲ 2 ▼ izhankhalib at gmail dot com 6 months ago

Below is our jsonform source code in mongo db which consists a lot of double quotes. we are able to pass this source code to the ajax form submit function by using php urlencode :

```
<script type="text/javascript">
$(function() {
    // Generate a form using jquery.dfrom
     $("#myform").dform({

     "html":[
        {
           "type":"p",
           "html":"Patient Record"
        },
        {
           "name":"patient.name.first",
           "id":"txt-patient.name.first",
           "caption":"first name",
           "type":"text",
        },
        {

           "name":"patient.name.last",
```

```
            "id":"txt-patient.name.last",
            "caption":"last name",
            "type":"text",
          },
          {
            "type" : "submit",
           }

      ]
   });
   });
</script>
<form id="myform">

<?php
//get the json source code from the mongodb
$jsonform= urlencode($this->data['Post']['jsonform']);

?>
//AJAX SUBMIT FORM
<script type="text/javascript">
$('#myform').submit(function(){


   //    passing the variable fro PHP to javascript
       var thejsonform="<?php echo $jsonform ?>";

 //var fname = $('input#fname').val();
 var dataString = "jsonform=" + thejsonform ;

   $.ajax({
        type: "POST",
       //  url: "test1.php",
        data: dataString,
        success: function() {

        }
       });


return false;
});
```

▲ 4 ▼ neugey at cox dot net 9 years ago

Be careful when encoding strings that came from simplexml in PHP 5. If you try to urlencode a simplexml object, the script tanks.

I got around the problem by using a cast.

$newValue = urlencode((string) $oldValue);

▲ 3 ▼ peter at runempire dot com 7 years ago

I think this was mentioned earlier but it was confusing.. But I had some problems with the urlencode eating my '/' so I did a simple str_replace like the following:

```
$url = urlencode($img);
$img2 = "$url";
```

```
$img2 = str_replace('%2F54', '/', $img2);
$img2 = str_replace('+' , '%20' , $img2);
```

You don't need to replace the '+' but I just feel comfortable with my %20, although it may present a problem if whatever you're using the str_replace for has a '+' in it where it shouldn't be.

But that fixed my problem.. all the other encodes like htmlentities and rawurlencode just ate my /'s

▲ 2 ▼ php dot net at unixen dot dk 5 years ago

>> Hi muthuishere , i saw your excellent contribution, but couldnt make it work, so i corrected some bits and pieces and had the following done:

```
<?php
function SmartUrlEncode($url){
   if (strpos($url, '=') === false):
      return $url;
   else:
      $startpos = strpos($url, "?");
      $tmpurl=substr($url, 0 , $startpos+1) ;
      $qryStr=substr($url, $startpos+1 ) ;
       $qryvalues=explode("&", $qryStr);
        foreach($qryvalues as $value):
           $buffer=explode("=", $value);
         $buffer[1]=urlencode($buffer[1]);
        endforeach;
       $finalqrystr=implode("&amp;", &$qryvalues);
      $finalURL=$tmpurl . $finalqrystr;
      return $finalURL;
   endif;
}
?>
```

As you see its very much yours, modfied primarily using '&' instead of '&', and ofcourse an if test to see if anything in input is to be cursed... Thanks for great function !

▲ 1 ▼ frx dot apps at gmail dot com 4 years ago

I wrote this simple function that creates a GET query (for URLS) from an array:

```
<?php
function encode_array($args)
{
  if(!is_array($args)) return false;
  $c = 0;
  $out = '';
  foreach($args as $name => $value)
  {
    if($c++ != 0) $out .= '&';
    $out .= urlencode("$name").'=';
    if(is_array($value))
    {
      $out .= urlencode(serialize($value));
    }else{
      $out .= urlencode("$value");
    }
  }
  return $out . "\n";
```

```
}
?>
```

If there are arrays within the $args array, they will be serialized before being urlencoded.

Some examples:

```
<?php
echo encode_array(array('foo' => 'bar'));                  // foo=bar
echo encode_array(array('foo&bar' => 'some=weird/value'));   // foo%26bar=some%3Dweird%2Fvalue
echo encode_array(array('foo' => 1, 'bar' =>  'two'));       // foo=1&bar=two
echo encode_array(array('args' => array('key' => 'value'))); //
args=a%3A1%3A%7Bs%3A3%3A%22key%22%3Bs%3A5%3A%22value%22%3B%7D
?>
```

1 Anonymous 5 years ago

What about this one? A bit more complex but very practically,...

```
<?php
static function urlencode($url)
{
      // safely cast back already encoded "&" within the query
      $url = str_replace( "&amp;","&",$url );
      $phpsep = (strlen(ini_get('arg_separator.input')>0))
?ini_get('arg_separator.output')
:"&";
      // cut optionally anchor
      $ancpos = strrpos($url,"#");
      $lasteq = strrpos($url,"=");
      $lastsep = strrpos($url,"&");
      $lastqsep = strrpos($url,"?");
      $firstsep = strpos($url, "?");
      // recognize wrong positioned anchor example.php#anchor?asdasd
      if ($ancpos !== false
      || ($ancpos > 0
         && ($lasteq > 0 && $lasteq < $ancpos )
         && ($lastsep > 0 && $lastsep < $ancpos )
         && ($lastqsep > 0 && $lastqsep < $ancpos )
         )
      )
      {
          $anc = "#" . urlencode( substr( $url,$ancpos+1 ) );
          $url = substr( $url,0,$ancpos );
      }
      else
      {
         $anc = "";
      }
      // separate uri and query string
      if ($firstsep == false)
      {
         $qry = "";    // no query
         $urlenc = $url.$anc;  // anchor
      }
      else
      {
         $qry = substr( $url, $firstsep + 1 ) ;
         $vals = explode( "&", $qry );
```

```
        $valsenc = array();
        foreach( $vals as $v )
        {
           $buf = explode( "=", $v );
           $buf[0]=urlencode($buf[0]);
           $buf[1]=urlencode($buf[1]);
           $valsenc[] = implode("=",$buf);
        }
        $urlenc = substr( $url, 0 , $firstsep  );    // encoded origin uri
        $urlenc.= "?" . implode($phpsep, $valsenc )    // encoded query string
        . $anc; // anchor
      }
      $urlenc = htmlentities( $urlenc, ENT_QUOTES );
      return $urlenc;
}
?>
```

▲ 1 ▼ kerdster at gmail dot com 6 years ago

> php dot net at samokhvalov dot com
> 12-Dec-2006 09:49

Thanx for idea!
I have wrote more simple function based on your function to simulate JS function escape (); It uses mb_string functions unstead of iconv.

```
<?php
function utf16urlencode($str)
{
   $str = mb_convert_encoding($str, 'UTF-16', 'UTF-8');
   $out = '';
   for ($i = 0; $i < mb_strlen($str, 'UTF-16'); $i++)
   {
      $out .= '%u'.bin2hex(mb_substr($str, $i, 1, 'UTF-16'));
   }
   return $out;
}
?>
```

▲ 1 ▼ R Mortimer 8 years ago

Do not let the browser auto encode an invalid URL. Not all browsers perform the same encodeing. Keep it cross browser do it server side.

▲ 1 ▼ sroolig at gmail dot com 2 years ago

I write this code, to full protect from XSS:

```
<?php
  $url = "";
  for($i = 0; $i < strlen($value); $i++)
   $url .= strpos("/:@&%=?.#", $value[$i]) === False ? urlencode($value[$i]) : $value[$i];
  $value = $url;
?>
```

▲ 0 ▼ homebot at yandex dot ru 9 months ago

Simple static class for array URL encoding

[code]

```php
<?php

/**
*
*  URL Encoding class
*  Use : urlencode_array::go() as function
*
*/
class urlencode_array
{

 /** Main encoding worker
 * @param string $perfix
 * @param array $array
 * @param string $ret byref Push record to return array
 * @param mixed $fe Is first call to function?
 */
 private static function encode_part($perfix, $array, &$ret, $fe = false)
 {
  foreach ( $array as $k => $v )
  {
   switch ( gettype($v))
   {
    case 'float'   :
    case 'integer' :
    case 'string'  : $ret [ $fe ? $k : $perfix.'['.$k.']' ] = $v; break;
    case 'boolean' : $ret [ $fe ? $k : $perfix.'['.$k.']' ] = ( $v ? '1' : '0' ); break;
    case 'null'    : $ret [ $fe ? $k : $perfix.'['.$k.']' ] = 'NULL'; break;
    case 'object'  : $v = (array) $v;
    case 'array'   : self::encode_part( $fe?$perfix.$k:$perfix.'['.$k.']' , $v, $ret, false); break;
   }
  }
 }

 /** UrlEncode Array
 * @param mixed $array Array or stdClass to encode
 * @returns string Strings ready for send as 'application/x-www-form-urlencoded'
 */
 public static function go($array)
 {
  $buff = array();
  if ( gettype($array) == 'object') $array = (array) $array;
  self::encode_part('', $array, $buff, true);
  $retn = '';
  foreach ( $buff as $k => $v )
   $retn .= urlencode($k) . '=' . urlencode($v) . '&';
  return $retn;
 }
}

#--------------------------------- TEST AREA -----------------------------------

$buffer = array(
 'master'  =>'master.zenith.lv',
 'join'    =>array('slave'=>'slave1.zenith.lv','slave2'=>array('node1.slave2.zenith.lv','slave2.zenith.lv')),
 'config'  => new stdClass()
);
```

```
$buffer['config']->MaxServerLoad  = 200;
$buffer['config']->MaxSlaveLoad   = 100;
$buffer['config']->DropUserNoWait = true;

$buffer = urlencode_array::go($buffer);
parse_str( $buffer , $data_decoded);

header('Content-Type: text/plain; charset=utf-8');
echo 'Encoded String :' . str_repeat('-', 80) . "\n";
echo $buffer;
echo str_repeat("\n", 3) . 'Decoded String byPhp :' . str_repeat('-', 80) . "\n";
print_r($data_decoded);

[/code]
```

▲ 0 ▼ davis dot peixoto at gmail dot com 3 years ago

urlencode function and rawurlencode are mostly based on RFC 1738.

However, since 2005 the current RFC in use for URIs standard is RFC 3986.

Here is a function to encode URLs according to RFC 3986.

```
<?php
function myUrlEncode($string) {
   $entities = array('%21', '%2A', '%27', '%28', '%29', '%3B', '%3A', '%40', '%26', '%3D', '%2B', '%24', '%2C', '%2F', '%3F', '%25',
'%23', '%5B', '%5D');
   $replacements = array('!', '*', "'", "(", ")", ";", ":", "@", "&", "=", "+", "$", ",", "/", "?", "%", "#", "[", "]");
   return str_replace($entities, $replacements, urlencode($string));
}
?>
```

▲ 0 ▼ temu92 at gmail dot com 4 years ago

I needed encoding and decoding for UTF8 urls, I came up with these very simple fuctions. Hope this helps!

```
<?php
   function url_encode($string){
      return urlencode(utf8_encode($string));
   }

   function url_decode($string){
      return utf8_decode(urldecode($string));
   }
?>
```

▲ 0 ▼ kL 7 years ago

Apache's mod_rewrite and mod_proxy are unable to handle urlencoded URLs properly - http://issues.apache.org/bugzilla/show_bug.cgi?id=34602

If you need to use any of these modules and handle paths that contain %2F or %3A (and few other encoded special url characters), you'll have use a different encoding scheme.

My solution is to replace "%" with "'".

```
<?php
function urlencode($u)
{
```

```
    return str_replace(array("'",'%'),array('%27',"'"),urlencode($u));
}

function urldecode($u)
{
    return urldecode(strtr($u,"'",'%'));
}
?>
```

0 roberlamerma at gmail dot com 8 years ago

I rewrote inus at flowingcreativity dot net function to generate an encoded url string from the POST, or GET array. It handles properly POST/GET array vars.

```
function _HTTPRequestToString($arr_request, $var_name, $separator='&') {
    $ret = "";
    if (is_array($arr_request)) {
        foreach ($arr_request as $key => $value) {
            if (is_array($value)) {
                if ($var_name) {
                    $ret .= $this->_HTTPRequestToString($value, "{$var_name}[{$key}]", $separator);
                } else {
                    $ret .= $this->_HTTPRequestToString($value, "{$key}", $separator);
                }
            } else {
                if ($var_name) {
                    $ret .= "{$var_name}[{$key}]=".urlencode($value)."&";
                } else {
                    $ret .= "{$key}=".urlencode($value)."&";
                }
            }
        }
    }
    if (!$var_name) {
        $ret = substr($ret,0,-1);
    }
    return $ret;
}
```

0 Bachsau 8 years ago

Diferrent from the above example you do not have to encode URLs in hrefs with this. The browser does it automaticaly, so you just have to encode it with htmlentities() ;)

0 monty3 at hotmail dot com 9 years ago

If you want to pass a url with parameters as a value IN a url AND through a javascript function, such as...

```
  <a href="javascript:openWin('page.php?url=index.php?id=4&pg=2');">
```

...pass the url value through the PHP urlencode() function twice, like this...

```
<?php

  $url = "index.php?id=4&pg=2";
  $url = urlencode(urlencode($url));

  echo "<a href=\"javascript:openWin('page.php?url=$url');\">";
```

```
?>
```

On the page being opened by the javascript function (page.php), you only need to urldecode() once, because when javascript 'touches' the url that passes through it, it decodes the url once itself. So, just decode it once more in your PHP script to fully undo the double-encoding...

```
<?php

  $url = urldecode($_GET['url']);
?>
```

If you don't do this, you'll find that the result url value in the target script is missing all the var=values following the ? question mark...

```
  index.php?id=4
```

▲ -1 ▼ Roamy: info AT roamy DOT de 8 years ago

```
<?// urlencode + urldecode 4 Linux/Unix-Servers:==============
//================================================
//=====This small script matches all encoded String for ========
//=====Linux/Unix-Servers For IIS it got to be  The Other Way  ==
//===== around...and remember in a propper Url ==============
//===== there shoudn't be the 'dirty Letter': %C3==============
//================================================
function int2hex($intega){
   $Ziffer = "0123456789ABCDEF";
return $Ziffer[($intega%256)/16].$Ziffer[$intega%16];
}
function url_decode($text){
   if(!strpos($text,"%C3"))
      for($i=129;$i<255;$i++){
         $in = "%".int2hex($i);
         $out = "%C3%".int2hex($i-64);
         $text = str_replace($in,$out,$text);
      }
return urldecode($text);
}
function url_encode($text){
   $text = urlencode($text);
   if(!strpos($text,"%C3"))
      for($i=129;$i<255;$i++){
         $in = "%".int2hex($i);
         $out = "%C3%".int2hex($i-64);
         $text = str_replace($in,$out,$text);
      }
return $text;
}//================================================
?>
```

▲ -1 ▼ torecs at sfe dot uio dot no 8 years ago

This very simple function makes an valid parameters part of an URL, to me it looks like several of the other versions here are decoding wrongly as they do not convert & seperating the variables into &.

```
 $vars=array('name' => 'tore','action' => 'sell&buy');
 echo MakeRequestUrl($vars);

 /* Makes an valid html request url by parsing the params array
  * @param $params The parameters to be converted into URL with key as name.
```

```
 */
function MakeRequestUrl($params)
{
   $querystring=null;
  foreach ($params as $name => $value)
  {
   $querystring=$name.'='.urlencode($value).'&'.$querystring;
  }
   // Cut the last '&'
   $querystring=substr($querystring,0,strlen($querystring)-1);
   return htmlentities($querystring);
}
```

Will output: action=sell%26buy&name=tore

▲ -1 ▼ linus at flowingcreativity dot net 8 years ago

I just came across the need for a function that exports an array into a query string. Being able to use urlencode($theArray) would be nice, but here's what I came up with:

```
<?php

function urlencode_array(
   $var,              // the array value
   $varName,          // variable name to be used in the query string
   $separator = '&'   // what separating character to use in the query string
) {
   $toImplode = array();
   foreach ($var as $key => $value) {
      if (is_array($value)) {
         $toImplode[] = urlencode_array($value, "{$varName}[{$key}]", $separator);
      } else {
         $toImplode[] = "{$varName}[{$key}]=".urlencode($value);
      }
   }
   return implode($separator, $toImplode);
}

?>
```

This function supports n-dimensional arrays (it encodes recursively).

▲ -2 ▼ bisqwit at iki dot fi 8 years ago

Constructing hyperlinks safely HOW-TO:

```
<?php
$path_component = 'machine/generated/part';
$url_parameter1 = 'this is a string';
$url_parameter2 = 'special/weird "$characters"';

$url = 'http://example.com/lab/cgi/test/'. rawurlencode($path_component) . '?param1=' . urlencode($url_parameter1) . '&param2=' .
urlencode($url_parameter2);

$link_label = "Click here & you'll be <happy>";

echo '<a href="', htmlspecialchars($url), '">', htmlspecialchars($link_label), '</a>';
?>
```

This example covers all the encodings you need to apply in order to create URLs safely without problems with any special characters. It is stunning how many people make mistakes with this.

Shortly:
- Use urlencode for all GET parameters (things that come after each "=").
- Use rawurlencode for parts that come before "?".
- Use htmlspecialchars for HTML tag parameters and HTML text content.

-1 Mark Seecof 5 years ago

When using XMLHttpRequest or another AJAX technique to submit data to a PHP script using GET (or POST with content-type header set to 'x-www-form-urlencoded') you must urlencode your data before you upload it. (In fact, if you don't urlencode POST data MS Internet Explorer may pop a "syntax error" dialog when you call XMLHttpRequest.send().) But, you can't call PHP's urlencode() function in Javascript! In fact, NO native Javascript function will urlencode data correctly for form submission. So here is a function to do the job fairly efficiently:

```
<?php /******

<script type="text/javascript" language="javascript1.6">
 // PHP-compatible urlencode() for Javascript
 function urlencode(s) {
  s = encodeURIComponent(s);
  return s.replace(/~/g,'%7E').replace(/%20/g,'+');
 }

 // sample usage:  suppose form has text input fields for
 // country, postcode, and city with id='country' and so-on.
 // We'll use GET to send values of country and postcode
 // to "city_lookup.php" asynchronously, then update city
 // field in form with the reply (from database lookup)

 function lookup_city() {
  var elm_country = document.getElementById('country');
  var elm_zip = document.getElementById('postcode');
  var elm_city = document.getElementById('city');
  var qry = '?country=' + urlencode(elm_country.value) +
            '&postcode=' + urlencode(elm_zip.value);
  var xhr;
  try {
   xhr = new XMLHttpRequest(); // recent browsers
  } catch (e) {
   alert('No XMLHttpRequest!');
   return;
  }
  xhr.open('GET',('city_lookup.php'+qry),true);
  xhr.onreadystatechange = function(){
    if ((xhr.readyState != 4) || (xhr.status != 200)) return;
    elm_city.value = xhr.responseText;
  }
  xhr.send(null);
 }
</script>

******/ ?>
```

-1 root at jusme dot org 5 years ago

I'm running PHP version 5.0.5 and urlencode() doesn't seem to encode the "#" character, although the function's description says it encodes "all non-alphanumeric" characters. This was a particular problem for me when trying to open local files with a "#" in the filename as Firefox will interpret this as an anchor target (for better or worse). It seems a manual str_replace is required unless this was fixed in a future PHP version.

Example:

$str = str_replace("#", "%23", $str);

-1 in reply to "kL" 7 years ago

kL's example is very bugged since it loops itself and the encode function is two-way.

Why do you replace all %27 through ' in the same string in that you replace all ' through %27?

Lets say I have a string: Hello %27World%27. It's a nice day.
I get: Hello Hello 'World'. It%27s a nice day.

With other words that solution is pretty useless.

Solution:
Just replace ' through %27 when encoding
Just replace %27 through ' when decoding. Or just use url_decode.

-4 tk at example dot com 3 years ago

To easily encode an array to an url :

```
<?php
$y= ("I"=>"like","she"=>"likes");
array_walk($y , create_function('&$v,$k', '$v = $k."=".$v ;'));
"index.php?".htmlentities(urlencode(implode("&",$y)), ENT_QUOTES);
?>
```

returns :
"index.php?I=like&she=likes"
=)

add a note

Fonctions URL

base64_decode
base64_encode
get_headers
get_meta_tags
http_build_query
parse_url
rawurldecode
rawurlencode
urldecode
» urlencode

Copyright © 2001-2014 The PHP Group My PHP.net Contact Other PHP.net sites Mirror sites

Privacy policy